

# Learning Radio Resource Management in 5G Networks: Framework, Opportunities and Challenges

Francesco D. Calabrese, Li Wang, Euhanna Ghadimi, Gunnar Peters, Pablo Soldati

Huawei Technologies Sweden AB, System Algorithms Lab, Kista, Sweden

e-mail: {firstname.lastname}@huawei.com

## Abstract

The fifth generation (5G) of mobile broadband shall be a far more complex system compared to earlier generations due to advancements in radio and network technology, increased densification and heterogeneity of network and user equipment, larger number of operating bands, as well as more stringent performance requirement. To cope with the increased complexity of the Radio Resources Management (RRM) of 5G systems, this manuscript advocates the need for a clean slate design of the 5G RRM architecture. We propose to capitalize the large amount of data readily available in the network from measurements and system observations in combination with the most recent advances in the field of machine learning. The result is an RRM architecture based on general-purpose learning framework capable of deriving specific RRM control policies directly from data gathered in the network. The potential of this approach is verified in three case studies and future directions on application of machine learning to RRM are discussed.

## Index Terms

machine learning, radio resource management, 5G, reinforcement learning, algorithm.

## I. INTRODUCTION

The next generation of mobile broadband (5G) systems will gather a variety of new services and features with stringent and rather diverse requirements in terms of peak data rate, reliability, latency and coverage [1]. High data rate services, such as enhanced Mobile Broadband (eMBB), will require seamlessly multi-connectivity across different Radio Access Technologies (RATs) operating over a wide range of frequency bands, ranging from below 6GHz to millimeter waves. Other services, such as Internet-of-Things (IoT), vehicular communication, etc. will be more sensitive to latency and reliability, requiring Ultra-Reliable Low-Latency Communications

(URLLC) and massive Machine Type Communications (mMTC). To meet such wide spectrum of requirements, 5G is expected to integrate a multiplicity of RATs into an increasingly complex and heterogeneous Radio Access Network (RAN) [1]–[3]. Optimizing Radio Resource Management (RRM) for such realm requires agile inter-RAT coordination as well as careful handling latency, reliability, quality of service and user experience constraints due to the coexistence of different traffic types. Therefore, only a complete redesign of the RRM architecture can fully cater to 5G systems. The RRM architecture of today's RANs comprises a maze of interconnected algorithms, each tailor-made for a specific RRM task. The design of such algorithms is strongly affected by two major aspects: the distortion of radio measurements (due to noise and delay) and the absence of clear relationships between such measurements and Key Performance Indicators (KPIs). As the optimization objective cannot be expressed in closed form as function of measurements and parameters, it is common practice to resort to simplified models that are unable to fully capture neither the complex dynamics of the system nor the variety of real-life scenarios. The resulting heuristics are designed with built-in robustness to work in a variety of scenarios at the expense of performance. Adding new features to the network typically requires new algorithms or redesigning existing ones, resulting into an increasingly fragmented and heterogeneous RRM architecture founded on an ever growing number of parameters. In recent years, a set of techniques, referred to as Self-Organizing Network (SON), were suggested to automate the tuning of such parameters [4]. While intended to address the growth of control parameters, SON ultimately resulted into an additional layer of rules (and parameters) on top of the legacy design. To avoid limitations and constraints of legacy designs, we advocate the need of a clean slate redesign of the 5G RRM architecture. While the RRM challenges are noticeable, the opportunities for a radically different RRM architecture are less obvious. One such opportunity is the variety of ways in which the radio environment is sensed and the extensive amount of data characterizing the environment available in RAN. Such data is made available by both the network and the user equipment, either in rather raw form, such as signatures of received power from reference signals at multiple antennae, or in processed forms, such as expressions of signal to noise ratio (and the like), bit and block error rates and other KPIs typically measured in the network. Additional knowledge is available to the network related to user mobility and traffic patterns, or more generally how/when/what the user does or needs in the network. Another opportunity is presented by the constantly growing computational capabilities of the network and user equipment which, combined with the centralized nature of the RAN architecture, offers a natural setup for

the application of Machine Learning (ML) techniques in RRM context. Concurrently, in the recent years, ML has experienced an unprecedented growth thanks to new techniques, more powerful computing tools and infrastructures capable of handling larger amounts of data in shorter time. Among these techniques, Reinforcement Learning (RL) is particularly suitable to solve complex control problems such as RRM in RAN. This paper proposes general-purpose 5G RRM architecture wherein a reinforcement learning framework generates and adapts control policies (i.e., algorithms) for different RRM tasks directly from data gathered in the network. Although RL has been tried successfully in different individual RRM tasks [5]–[7], the application of RL to complex systems, such as a full-scale RAN, poses several challenges. While highlighting such challenges, we suggest efficient ways to resolve them and demonstrate the effectiveness of the framework in a number of case studies.

## II. A BRIEF OVERVIEW OF LEARNING METHODS

Machine Learning is a discipline that deals with the task of inferring a function (or pattern) from a set of noisy data (known as training set) generated by an unknown true function. Different categories of ML algorithms exist depending on the nature of the learning problem. In the context of this paper, the branches of interest are Supervised Learning and Reinforcement Learning. Supervised Learning (SL) infers a function from a set of data pairs consisting of an input and a corresponding desired output (or label), generally vectors, provided by a supervisor. Typical applications of SL include regression and classification problems. Since the data samples are typically noisy and the training set only represents a fraction of the complete data set, principles and techniques exist to mitigate the dangers of fitting the noise, such as regularization and k-fold cross-validation [8]. A plethora of techniques exists to recover an unknown function from labeled data. One such method, which today experiences a significant revival, is Artificial Neural Networks (ANNs). ANNs exist in a variety of architectures which are typically tailored to the task at hand. Despite their biological inspiration, ANNs are essentially a network of sums and products whose linear combinations are passed through a squashing non-linear function and whose weights are adjusted (cf. trained) so as to produce a function which should be as close as possible to the true function that produced the input-output pairs of the training set. Training an ANN takes place by gradually adjusting its weights in the direction of the gradient, with respect to the weights, of the expected error loss function between the function represented by the ANN and the actual noisy samples (the input-output pairs) produced by the original true function.

The application of this gradient rule, specialized to ANN, is known as back-propagation [9]. Reinforcement Learning (RL) is an area of ML dealing with how a software agent learns to behave in a given environment in order to achieve a given objective, such as maximizing a form of reward. RL has been found particularly suitable for large-scale control problems [9]. Unlike a model-based setting, which attempts to recover a model of the environment, we consider a model-free setup, where the problem is described uniquely in terms of three components: state, action and reward. The state  $s$  is a tuple of values, known as *features*, that describes the agent in relation to the environment in a way that is relevant for the problem at hand. The action  $a$ , chosen in a set of available actions, represents the change that the agent applies to the environment in order to maximize the given form of reward. The reward  $r$  is a multi-objective scalar function which numerically expresses what the agent should achieve. The interaction, over time, of the agent with the environment is captured by a set of transitions consisting of going from a state to the next state by applying actions to the environment and receiving rewards as a consequence of such actions. Each transition is a quadruple of the form  $(s_t, a_t, r_{t+1}, s_{t+1})$  where  $t$  is a discrete time counter. The first objective of RL is to extract, from the transitions set, a policy  $\pi$  that, given a state, returns the action to take in order to maximize the long-term cumulative reward. The RL algorithm thus needs to map the rewards to the actions, possibly taken far back in time, that lead to such rewards. This notion is known as *credit-assignment* [9]. The second objective of an RL framework is that of rapidly bringing the agent from a tabula-rasa state, where he does not know how to act, to a condition where it is acting as close to optimality as possible. Making as few mistakes as possible in the path to a quasi-optimal behavior is known as *regret minimization*, a notion closely related to the topic of trading off *exploration* of the environment (to sample unseen parts of the state-action space at the cost of not choosing the best known action in that state) with *exploitation* of the knowledge accumulated so far (to maximize the reward at the cost of not trying a new potentially better action) [9]. This gradual transition from a pure exploration strategy to an exploitation strategy can be implemented using a variety of techniques, e.g.  $\epsilon$ -greedy algorithm [9].

### III. ENABLING RL FOR 5G RRM

RRM in modern RANs is essentially a combination of several control problems, each pertinent to controlling a set of radio parameters or a set of decisions to optimize suitable combinations

of KPIs. As such, RL offers a powerful alternative to redesign RRM algorithms in complex and highly dynamic systems such as future 5G RANs.

### *A. Challenges*

Basic formulations of RL have recently been suggested to address specific RRM problems in the wireless communication literature, see e.g. [5]–[7], where the learning algorithm exploits tables to store a running average of a learned functional value for each state-action pair. As the number of possible states and actions grows, the number of data samples required to train this type of algorithm becomes prohibitively large. Moreover the accumulated experience cannot be generalized across states that are similar with each other. More generally, the application of RL to a full-scale RAN poses challenges that these basic RL algorithms cannot entirely address. The first challenge faced in a control problem of this kind is its large dimensionality. The variety of conditions in the network paired with the number of parameters changes that could be made, results in an extremely large state-action space. This problem becomes particularly severe in 5G systems which are expected to support massive antenna arrays and massive number of connections, wider spectrum bands, latency-critical and ultra-reliable applications, as well as to enable flexible reconfiguration of the air interface possibly with different numerology in different spectrum sub-bands (e.g., flexible sub-frame length and sub-carrier spacing) [1]–[3]. Two additional challenges for applying RL to RAN are the multi-agent nature of RANs and the partial observability of the network state. As a result, the response that one agent receives from the environment is likely to be the consequence of the actions of other agents as well. On the other hand, each agent, taken individually, has access to a reduced part of the full RAN state, such as local measurements taken by users in a cell or by the radio base station. Addressing these challenges requires a framework that enables efficient inter-agent coordination and communication mechanisms to exchange information about the local state and reward. Another complication in RANs is that the evolution of the environment observed by the agent is stochastic, i.e. the application of actions to the environment does not result in a deterministic outcome (like it would be, for example, in a board game). This is largely the consequence of the previous two issues, that is, the partial observability of the state and the presence of multiple agents acting at the same time. Finally, a challenge of practical nature is to ensure a robust system performance under safety constraints during both exploration and exploitation processes so as to prevent system failure or a prolonged degradation of the system performance. For instance, risk-averse criteria could be

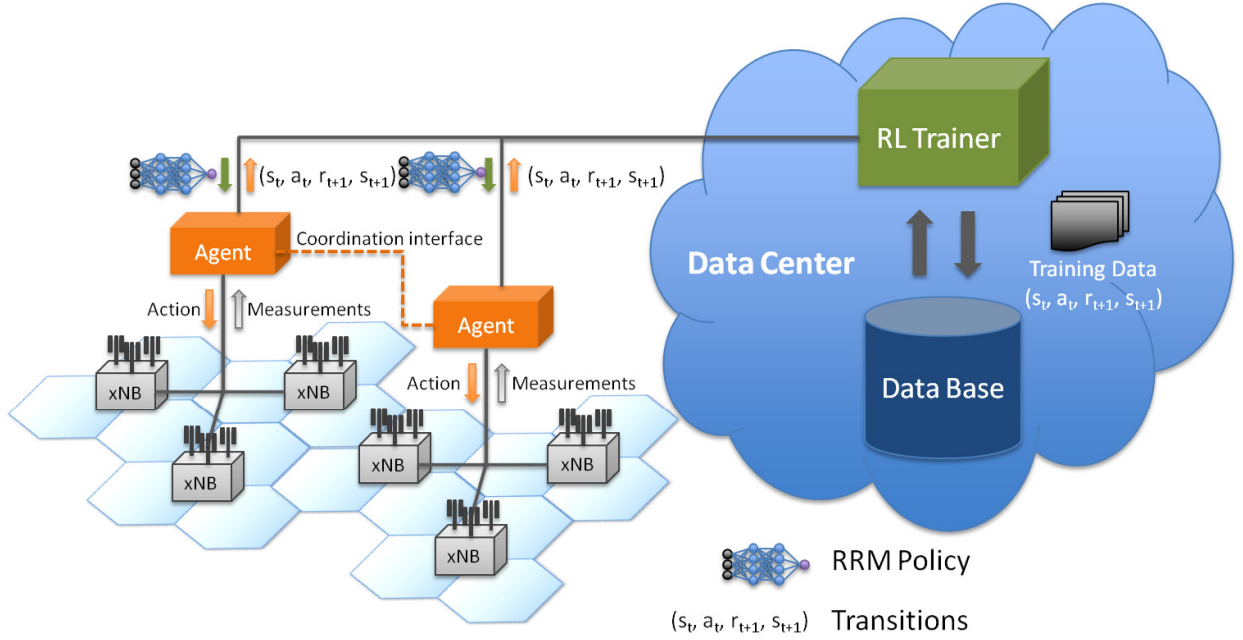


Figure 1: RRM architecture envisioned to enable RL applied to RANs

integrated in different components of the RL framework, e.g. the exploration strategy and the reward definition.

### B. Opportunities: A general-purpose 5G RRM architecture

Machine learning, and RL in particular, offers an opportunity to radically rethink the design of the RRM architecture for 5G RAN by shifting the engineering effort from designing numerous, single purpose, RRM algorithms - as legacy RRM architectures - to designing and improving a single, general purpose, learning framework. In this vision, the same learning principle (i.e., the same algorithm) is used to generate specialized control policies for individual RRM tasks executed by network entities. Figure 1 illustrates the basic components of the 5G RRM architecture required to realize this vision. In this context, the RAN represents the environment; the state, in its entirety, can be represented by a set of features characterizing the agent in relation to the network, such as the type and number of terminals, their traffic, their positions, their capabilities, the type and number of cells, the different measurements and KPIs (e.g., cell coverage, cell capacity, packet delay, etc.). In practice a more synthetic representation formed by a subset of features is used. The

actions are represented by parameters adjustments (e.g., power-up, power-down, power-hold); the reward signal can be a function (not necessarily linear) of the KPIs.

The trainer and the agent are the key (logical) components of the proposed RRM architecture. The trainer applies a single learning algorithm to derive specialized control policies for different agents and RRM tasks. The agent executes the policies issued by the trainer to interact with the network. While typically such two components are treated as one single entity, the design challenges related to the RAN suggest that the trainer, where policies are derived, and the agent, where policies are executed, should be residing in different network entities as suggested in Figure 1. The trainer could reside in a data center, e.g. a server farm or, in a smaller scale, a cloud RAN. Transitional data received from the agents is collected and stored by the trainer for deriving or updating policies tailored to specific RRM tasks and agents (cf. Sec. IV). This has the advantage of concentrating the computational complexity and data storage in a single place while enabling techniques for greater data efficiency (cf. Sec. V). The location of the agent may depend on deployments and scenarios. In a multi-RAT system, as in Figure 3a, an agent can reside in the baseband of a cluster of co-located radio cells, whereas an agent controlling non co-located RATs may reside at higher hierarchical levels, e.g., within a Mobility Management Entity (MME), within the serving gateway, etc.

#### IV. RL ALGORITHMS DESIGN FOR 5G RRM

The strength and flexibility of the proposed architecture resides in the learning algorithm chosen to generate specialized control policies for individual RRM tasks. Hereafter we discuss the trainer and agent algorithm design.

##### A. General purpose RRM trainer design

The literature of machine learning offers several types of RL algorithms for deriving policies in the trainer. One such algorithm that proved to be very effective is Q-learning [10], which aims at learning an action-value function (the Q-function) to estimate a long-term reward. Given a policy  $\pi$ , the Q-function is the expected utility of taking an action  $a$  in a given state  $s$  and following the policy  $\pi$  afterwards. That is

$$Q_{\pi}(s_t, a_t) = \mathbb{E}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots],$$

where  $\gamma \in [0, 1)$  is a discounting factor causing the value of rewards to decay exponentially over time, thus adjusting the preference for immediate rewards while making the optimization

horizon for the agent finite (that is, the sum of rewards is finite). Larger values of  $\gamma$  allow us to optimize for longer-term rewards thus improving the KPIs of the system.

In the attempt to maximize the long-term reward, RL searches a policy that maximizes the Q-function. Since a wide range of RL problems (including the ones of RRM) can be modeled as Markov decision processes, such an optimum Q-function can be found as the solution to the Hamilton-Jacobi-Bellman equation [9]:

$$Q(s_t, a_t) = r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}).$$

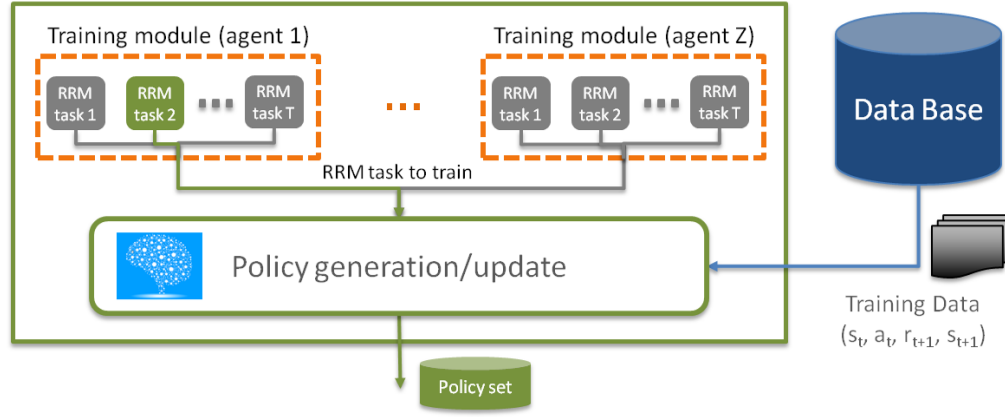
We therefore adopt a Q-learning approach to derive RRM policies at the trainer, which includes deriving a policy that optimizes the Q-function. Contrary to the traditional approach of relying on large lookup table of Q-functions associated to each state-action pairs, we propose to combine Q-learning with functional approximation of the Q-function [9]. Specifically, instead of updating the Q-function in individual entries of a lookup table, the parameters of a functional approximation are changed so as to minimize the error between the predicted Q-value  $Q(s_t, a_t)$  and the target value  $r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$ . In addition to accelerating the learning, this design choice enables to generalize across states (thus also to previously unseen states).

The proposed learning framework can exploit different functional approximation methods, such as ANNs and decision forests. However, we present a description of the training algorithm based on ANNs since their proved efficiency in compressing the large-dimensional inputs into low-dimensional outputs makes them one of the preferred choices of non-linear function approximations. While there exist a variety of ways in which ANNs can be used in an RL framework, arguably, one attractive frameworks in the machine learning literature combining ANNs and Q-learning is the Neural-Fitted Q Iteration (NFQ) algorithm [11]–[13]. With NFQ, the policy generation procedure consists of two loops: an outer loop where the inputs  $(s_t, a_t)$  and outputs  $(r_{t+1} + \gamma \max_{a_{t+1}} Q_{k-1}(s_{t+1}, a_{t+1}))$  are generated based on the current ANN ( $Q_{k-1}$ ), and an inner loop where the weights of ANN are adjusted to fit the Q-function to input-output training pairs ( $Q_{k-1} \rightarrow Q_k$ ) (see Figure 2a, 2b).

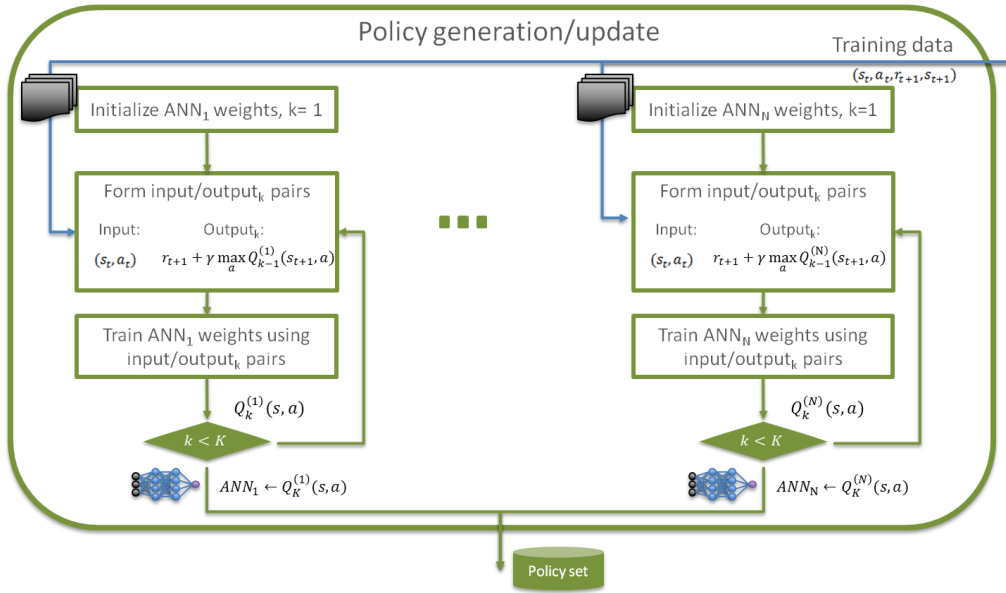
## B. RRM Agent design

The RRM agent receives updated control policies for different RRM tasks from the trainer and interacts with the underlying network by executing such policies. Upon taking an action associated with an RRM task, the agent gathers observations of the network in the form of L1/L2





(a) RRM trainer



(b) Policy generation procedure

Figure 2: Illustration of functional blocks of a general purpose RRM trainer design (a) and policy generation procedure based on Q-learning and NFQ for an ensemble of ANNs (b).

measurements and other KPIs. Measurements are processed into features relevant to characterize the state associated to each RRM task, and transitions of the form  $(s_t, a_t, r_{t+1}, s_{t+1})$  are sent to the trainer, cf. Figure 3a.

In this example, the agent receives a policy update in the form of weights of ANNs. The use of multiple ANNs with distinct structures and configurations (e.g., number of layers and neurons per layer) independently trained in the same way to learn the Q-function, as in Figure 2b, is known as *ensemble learning*. As long as each ANN is correct more than 50% of the time and the ANNs of



## V. LEARNING EFFICIENCY

Wireless networks are an always running critical system. Learning in an efficient way is therefore crucial to minimize the impact of the exploration while retaining all the advantages of a self-learned control policy. Hereafter we discuss four approaches.

### A. *Efficient data usage*

One approach is to have a better use of the available data through an effective credit assignment algorithm, such as the NFQ in conjunction with batch training advocated in our framework. Contrary to pure online Q-learning, with batch-training Q-learning the state transition tuple  $(s_t, a_t, r_{t+1}, s_{t+1})$  are stored and reused during the batch training in order to spread the information available through the whole state space.

### B. *High quality data collection*

The approach is to improve the quality of the data via more advanced exploration schemes which direct the exploration towards those parts of the state-action space that provide richer information to achieve optimal policies. Embedding human expert knowledge can help confining the exploration within the most relevant regions of the learning space and re-visit such regions to gather more informative data samples.

### C. *Compact state representation*

The third strategy to facilitate the learning task is to reduce the number of features representing the state in order to reduce the dimensionality of the learning space. This requires that we carefully choose or design features that are as informative as possible of the state of the agent in relation to the RRM problem of interest.

### D. *Transfer learning*

Improving the learning efficiency can also be done by “parameter transfer” and/or “instance transfer”. The former consists in sharing the same policy (e.g. in terms of ANN weights) between different agents. This is particularly useful, for instance, at deployment stage, when instead of starting from a tabula-rasa agent, the newly deployed equipment can be upgraded with policies learned for other agents. The latter consists in sharing experience (e.g., the transition samples)

among agents in the network. The architecture in Figure 1 encourages this strategy by gathering transitions from all agents at the trainer and using the collective experience of the agents, rather than from the experience of a single agent, to derive policies. A variant of the instance transfer approach consists in providing the agents with artificial samples generated from a simulator which allows to jump-start the learning of the agents. Given that the simulated models and the reality are different, exploration is still required in order to adapt the policy to the real world.

## VI. EVALUATION STUDY CASES

We demonstrate the generality of the proposed learning framework in three RRM study cases in a LTE-A compliant event driven simulator and show significant performance gains compared to traditional schemes.

### A. *Distributed power control*

RAN densification is one of the key enablers to meet the ever increasing demands for higher data rates and coverage in future cellular networks [1]. Effective inter-cell interference coordination plays a critical role in the success of such technologies. We first apply the RL framework for distributed inter-cell interference coordination via power control and rate adaptation in the downlink of radio access networks. Figure 4 shows the convergence of RL based algorithm on a 2-cell/2-agents example with full-buffer traffic and randomly distributed users. Agents take turns every 100 milliseconds to control the power budget of their own cell. The network harmonic mean throughput and the network sum-log throughput are considered as reward functions. Starting from an initial exploration probability of 90%, agents extensively explore the state-action space while gradually decreasing the exploration probability until reaching the final value of 10% within 40 seconds. The figure shows that agents successfully learn power control policies optimizing toward the associated rewards. More extensive simulations with uneven loads and burst data traffics show that the RL framework driven by a harmonic mean throughput reward is quickly able to produce a power control policy which results in energy saving and fairness across users [15]. In particular, compared to a fixed power budget baseline, our scheme achieves a throughput gain of 94% and 22% for 5%-tile and the median users, respectively. Moreover, it achieves 90% power saving per cell.

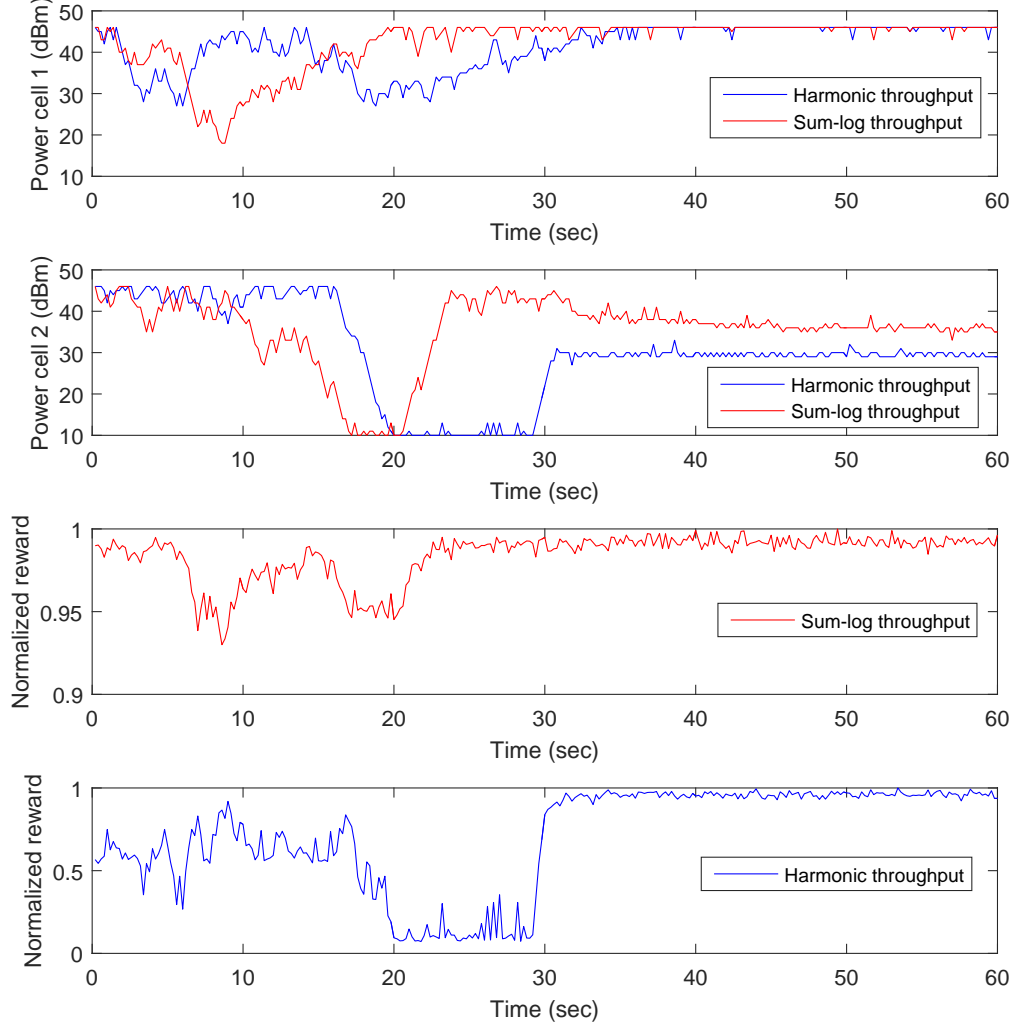


Figure 4: The convergence of RL based power control policy in a 2-agent use-case.

### B. Load Balancing in dense HetNets

Providing seamless hand-offs and increased mobile data capacity by integration of high power macro with low power pico cells forming a Heterogeneous Network (HetNet), as envisioned for future RANs, comes at the expense of creating more *cell-edges*. In particular, the difference between the transmit power budgets of pico and macro cells causes unevenness in cell loads and renders the interference management more challenging compared to homogeneous networks.

To address the load balancing problem experienced by the macro-pico HetNet, we used RL framework to control pico's Cell Individual Offset (CIO) value within the 3GPP Cell Range Extension (CRE) scheme. At each pico BS, an RL agent adjusts the CIO values via quantized values (e.g.  $\pm 1$  dB, or 0 dB) and informs its UEs. Then, the notified UEs update a table containing

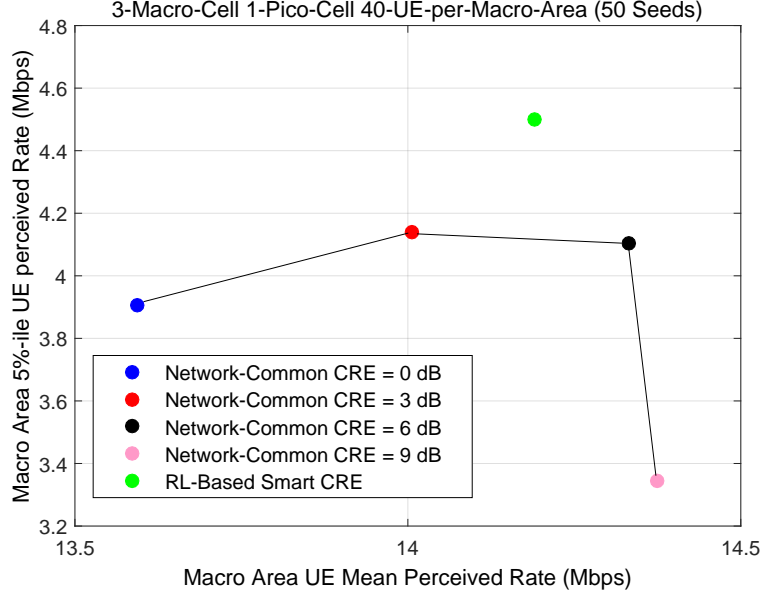


Figure 5: Performance gain achieved by reinforcement learning based CRE-aided load balancing.

all the neighboring pico's CIOs. Sufficiently large CIO adjustments may lead into a handover between a macro and pico cell. Figure 5 shows that the RL based CRE load balancing scheme with the cell-edge UE perceived data rate as the reward is able to strike a better balance between coverage and capacity compared to the traditional schemes with different fixed network-common CIO configurations.

### C. Smart ASFN

Single-frequency network (SFN) is an effective transmit-diversity technique to eliminate the traditional cell edges, thus improving the coverage, for both indoor and outdoor users. With SFN, a group of remote radio units (RRUs) form a virtual cell transmitting the same data to a given UE on the same time-frequency resources. While transmit diversity via such joint transmission improves the coverage, the network capacity is typically deteriorated. To improve capacity without losing coverage, a fixed signal-to-interference ratio (SIR) threshold can be used to select a set of serving RRUs associated to UE so that users with non-overlapping RRU sets can be co-scheduled. While this enhanced SFN (eSFN) improves frequency reuse, it also increases interference at the expense of peak rates. To truly boost the network capacity, the RRU selection needs to be adaptive taking into account the time-varying user/system-load distribution. Since a proper model of these complex system interactions cannot be designed, we devise an RL based adaptive SFN

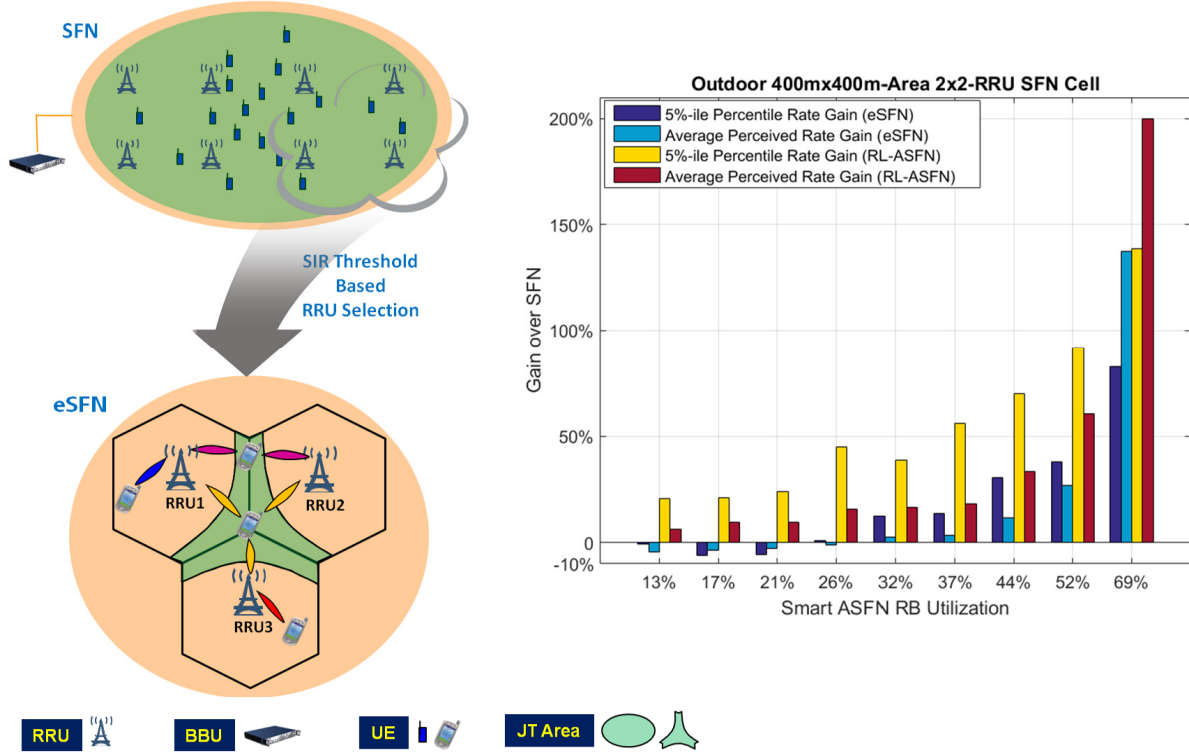


Figure 6: RL based smart ASFN performance gain.

(RL-ASFN) that dynamically optimizes the SIR threshold and enables the following built-in functionalities:

- Awareness of time-varying network load geographical distribution;
- High adaptability to the changes of network deployment through self-learning;
- Self-optimization with fine granularity realizing per-RRU joint-transmission area breathing.

Figure 6 shows the significant performance gains in coverage and capacity achieved by RL-ASFN with time-varying system load (i.e. RB utilization). Simulations are based on real network traffic in Singapore with small and big packet UEs with Poisson distribution arrivals. In comparison to the eSFN with fixed network-common SIR threshold configuration, RRUs in RL-ASFN cooperatively learn to adapt their threshold settings for time-varying load distributions in order to maximize the cell harmonic mean UE perceived rate.

## VII. FINAL REMARKS AND FUTURE DIRECTIONS

Our vision to achieve flexible and agile RRM in 5G is a clean-slate RRM architecture design based on a general purpose learning framework capable of autonomously generating control policies (i.e., algorithms) to solve complex and highly dimensional RRM tasks directly from data gathered in the network. In this vision, the engineering effort is shifted from designing numerous, single purpose, RRM algorithms - as legacy RRM architectures - to designing and improving a single, general purpose, learning framework. Experience (i.e., data) gathered by access nodes in the operator's network can be reused to either generate RRM policies at network deployment or to update individual policies to track changes in the environment. This creates the opportunity to more effectively deploy/upgrade network equipment, thereby reducing CAPEX and OPEX for the operator while enhancing the system performance. While our initial approach considers Reinforcement Learning with Neural-Fitted Q-learning, the fast pace of advancements in the field of machine learning regularly makes available new and powerful techniques that can enrich the proposed RRM architecture. A natural extension of this work is to enhance the learning framework to solve jointly multiple RRM tasks with a single learned policy. Broadening the task solved by the learning framework comes at the price of increased complexity and dimensionality, which requires not only more data to train the control policy but could also benefit from more refined learning techniques. This generalization is at the core of any future investigation considering machine learning for solving RRM problems.

## REFERENCES

- [1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, June 2014.
- [2] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Communications Magazine*, vol. 2, pp. 74–80, Feb. 2014.
- [3] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture," *IEEE Communications Magazine*, vol. 11, pp. 65–77, Nov. 2014.
- [4] A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5G: How to empower SON with Big Data for enabling 5G," *IEEE Networks*, vol. 6, pp. 27–33, Nov. 2014.
- [5] A. Galindo-Serrano and L. Giupponi, "Distributed Q-learning for interference control in OFDMA-based femtocell networks," in *IEEE Vehicular Technology Conference*, 2010.
- [6] M. Bennis and D. Niyato, "A Q-learning based approach to interference avoidance in self-organized femtocell networks," in *IEEE Globecom Workshops*, 2010.



- [7] I. Macaluso, D. Finn, B. Ozgul, and L. A. DaSilva, "Complexity of spectrum activity and benefits of reinforcement learning for dynamic channel selection," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 11, pp. 2237–2248, Nov. 2013.
- [8] Y. S. Abu-Mustafa, M. Magdon-Ismail, and H.-T. Lin, *Learning from Data - A short Course*. AML Book, 2012.
- [9] S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.
- [10] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, University of Cambridge, England, 1989.
- [11] M. Riedmiller, *Neural fitted Q Iteration - First experiences with a data efficient neural reinforcement learning method*. Springer, 2005.
- [12] T. Gabel, C. Lutz, and M. Riedmiller, "Improved neural fitted Q iteration applied to a novel computer gaming and learning benchmark," in *IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning, ADPRL*, April 2011.
- [13] M. Riedmiller and H. Braun, "A direct adaptive method for faster back propagation learning: The RPROP algorithm," in *IEEE Int. Conf. on Neural Networks*, 1993.
- [14] C. Gehring and D. Precup, "Smart exploration in reinforcement learning using absolute temporal difference errors," in *the 12th International Conference on Autonomous Agents and Multiagent Systems*, 2013.
- [15] E. Ghadimi, F. D. Calabrese, G. Peters, and P. Soldati, "A reinforcement learning approach to power control and rate adaptation in cellular networks," *Eprint arXiv:1611.06497*, 2016.